

1       1. A boot method for an In-Circuit Emulation system having a microcontroller  
2       operating in lock-step synchronization with a virtual microcontroller, comprising:

3               in the microcontroller, executing a set of boot code;

4               in the virtual microcontroller, executing a set of timing code timed to take the  
5       same number of clock cycles as the microcontroller uses to execute the boot code;  
6       and

7               simultaneously halting both the microcontroller and the virtual  
8       microcontroller.

9  
10      2. The method according to claim 1, further comprising copying register  
11       contents from the microcontroller to corresponding registers in the virtual  
12       microcontroller.

13  
14      3. The method according to claim 1, further comprising copying memory  
15       contents from the microcontroller to corresponding memory in the virtual  
16       microcontroller.

17  
18      4. The method according to claim 1, wherein after the executing of the boot  
19       code, the microcontroller branches to assembly instruction line 0; and wherein after  
20       executing the timing code, the virtual microcontroller branches to assembly  
21       instruction line 0.

22  
23      5. The method according to claim 1, wherein prior to the executing of the boot  
24       code, and prior to executing the timing code, a break is set at assembly instruction  
25       line 0.

26  
27      6. The method according to claim 1, wherein the boot code is stored within the  
28       microcontroller and hidden from the virtual microcontroller.

1       7. The method according to claim 1, further comprising:  
2               prior to the executing of the boot code, and prior to executing the timing  
3               code, setting a break at assembly instruction line 0; and  
4               wherein after the executing of the boot code, the microcontroller branches  
5               to assembly instruction line 0; and wherein after executing the timing code, the  
6               virtual microcontroller branches to assembly instruction line 0.

7  
8       8. The method according to claim 1, further comprising:  
9               prior to the executing of the boot code, and prior to executing the timing  
10               code, setting a break at assembly instruction line 0;  
11               wherein after the executing of the boot code, the microcontroller branches  
12               to assembly instruction line 0; and wherein after executing the timing code, the  
13               virtual microcontroller branches to assembly instruction line 0;  
14               copying register contents from the microcontroller to corresponding registers  
15               in the virtual microcontroller;  
16               copying memory contents from the microcontroller to corresponding memory  
17               in the virtual microcontroller;  
18               wherein after the executing of the boot code, the microcontroller branches  
19               to assembly instruction line 0; and  
20               wherein after executing the timing code, the virtual microcontroller branches  
21               to assembly instruction line zero.

22  
23       9. The method according to claim 8, further comprising removing the break at  
24               assembly line zero after copying the register contents and copying the memory  
25               contents.

1 10. A boot method for an In-Circuit Emulation system having a microcontroller  
2 operating in lock-step synchronization with a virtual microcontroller, comprising:  
3       resetting the microcontroller and the virtual microcontroller to a halt state;  
4       setting a break at assembly instruction line zero;  
5       in the microcontroller, executing a set of boot code;  
6       in the virtual microcontroller, simultaneously executing a set of timing code  
7 timed to take the same number of clock cycles as the microcontroller uses to  
8 execute the boot code;  
9       simultaneously halting both the microcontroller and the virtual  
10 microcontroller by branching to assembly instruction line zero;  
11       copying register contents from the microcontroller to corresponding registers  
12 in the virtual microcontroller;  
13       copying memory contents from the microcontroller to corresponding memory  
14 in the virtual microcontroller; and  
15       removing the break at assembly line zero after copying the register contents  
16 and copying the memory contents.  
17

18 11. The method according to claim 10, wherein the boot code is stored within  
19 the microcontroller and hidden from the virtual microcontroller.

1       12. A boot method for an In-Circuit Emulation system having a device under test  
2       operating in lock-step synchronization with a virtual processor, comprising:

3               in the device under test, executing a set of boot code;  
4               in the virtual processor, executing a set of timing code timed to take the  
5       same number of clock cycles as the device under test uses to execute the boot  
6       code; and  
7               simultaneously halting both the device under test and the virtual processor.

8

9       13. The method according to claim 12, further comprising copying register  
10      contents from the device under test to corresponding registers in the virtual  
11      processor.

12

13       14. The method according to claim 12, further comprising copying memory  
14      contents from the device under test to corresponding memory in the virtual  
15      processor.

16

17       18. The method according to claim 12, wherein after the executing of the boot  
18      code, the device under test branches to assembly instruction line 0; and wherein  
19      after executing the timing code, the virtual processor branches to assembly  
20      instruction line 0.

21

22       23. The method according to claim 12, wherein prior to the executing of the boot  
23      code, and prior to executing the timing code, a break is set at assembly instruction  
24      line 0.

25

26       27. The method according to claim 12, wherein the boot code is stored within  
27      the microcontroller and hidden from the virtual microcontroller.

1       18. The method according to claim 12, further comprising:  
2               prior to the executing of the boot code, and prior to executing the timing  
3               code, setting a break at assembly instruction line 0; and  
4               wherein after the executing of the boot code, the device under test branches  
5               to assembly instruction line 0; and wherein after executing the timing code, the  
6               virtual processor branches to assembly instruction line 0.

7  
8       19. The method according to claim 12, further comprising:  
9               prior to the executing of the boot code, and prior to executing the timing  
10               code, setting a break at assembly instruction line 0;  
11               wherein after the executing of the boot code, the device under test branches  
12               to assembly instruction line 0; and wherein after executing the timing code, the  
13               virtual processor branches to assembly instruction line 0;  
14               copying register contents from the microcontroller to corresponding registers  
15               in the virtual processor;  
16               copying memory contents from the device under test to corresponding  
17               memory in the virtual microcontroller;  
18               wherein after the executing of the boot code, the microcontroller branches  
19               to assembly instruction line 0; and  
20               wherein after executing the timing code, the virtual processor branches to  
21               assembly instruction line zero.

22  
23       20. The method according to claim 19, further comprising removing the break  
24               at assembly line zero after copying the register contents and copying the memory  
25               contents.

26  
27       21. The method according to claim 12, wherein the virtual processor is  
28               implemented in a field programmable gate array.